THIS OPINION WAS NOT WRITTEN FOR PUBLICATION

The opinion in support of the decision being entered today (1) was not written for publication in a law journal and (2) is not binding precedent of the Board.

Paper No. 17

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte WILLIAM R. BRYG,
KENNETH K. CHAN, ERIC DELANO,
and JOHN F. SHELTON

Appeal No. 1997-0708
Application 08/196,618[1]

ON BRIEF

Before BARRETT, FLEMING, and FRAHM, Administrative Patent Judges.

BARRETT, Administrative Patent Judge.

DECISION ON APPEAL

[1] Application for patent filed February 14, 1994, entitled (as amended in Paper No. 5) "Method And Apparatus For Checking Cache Coherency In A Computer Architecture."

This is a decision on appeal under 35 U.S.C. § 134 from the final rejection of claims 1, 2, 4-7, 9, and 10. Claims 3 and 8 stand objected to as being dependent upon a rejected base claim.

We reverse.

BACKGROUND

The disclosed invention is directed to a method and apparatus for checking cache coherency. "Snoop" refers to monitoring bus traffic to maintain cache coherency. When a main memory bus transaction occurs to an address which is replicated in the cache, a snoop hit is detected and appropriate actions are taken. The invention provides a double cache snoop mechanism, i.e., a cache mechanism in which each snoop has the possibility of being sent to the cache twice, first a read-only request to determine if there is a hit and, if so, a read-write request to modify the cache. This reduces the average number of cycles that a processor is stalled or locked during a coherency check.

Claim 1 is reproduced below.

> 1. An apparatus for checking cache coherency in a computer architecture having a system memory interconnected by a system bus to at least two modules,

at least one of said modules having an associated cache memory for storing cache contents therein, comprising:

a cache address tag associated with each cache for indicating said cache contents status;

a bus interface associated with each module for receiving requests to check cache coherency, and for forwarding said requests to said cache; said bus interface forwarding a first, read-only request to said cache, followed by a second read-write request that is forwarded to said cache if and only if said first read-only request indicates that said cache address tag matches an address of said request to check cache coherency and said request to check cache coherency requires said cache to be modified as a result of any of invalidation, copying out, or changing from clean to shared status; and

each module including a lock for interrupting access to said cache only during a read-only request and a subsequent read-write request pursuant to a cache coherency check.

The Examiner relies on the following prior art:

Thacker 5,136,700 August 4, 1992

Claims 1, 2, 4-7, 9, and 10 stand rejected under 35 U.S.C. § 103 as being unpatentable over Thacker.

We refer to the Final Rejection (Paper No. 7) (pages referred to as "FR__") and the Examiner's Answer (Paper No. 14) (pages referred to as "EA__") for a statement of the Examiner's position and to the Corrected Appeal Brief filed

May 16, 1996, (Paper No. 13) (pages referred to as "Br__") for Appellants' arguments thereagainst.

OPINION

Appellants group claims 1-5 together in one group and claims 6-10 together in another group (Br11). However, since the claims are not argued individually, the claims will be treated as standing or falling together with claim 1.

Appellants argue that nothing in Thacker's two-level cache mechanism renders Appellants' double snoop mechanism obvious to one of ordinary skill in the art. We agree. While it is sometimes possible for claims to be interpreted broadly to read on the prior art in a manner not anticipated by or intended by an applicant, this is not such a case.

The obviousness rejection is based on a single reference. Therefore, we would expect the differences between Thacker and the claimed subject matter to be very slight and of the kind that would be within the well known knowledge of one of ordinary skill in the art of cache design. Thacker is directed to cache checking and does have a cache structure with tags, but otherwise does not come close to meeting the claimed subject matter.

The Examiner finds that Thacker discloses "a bus interface [264 and 274] associated with each module [120 and 122] for receiving requests to check [Comp 262 or 272]" (FR2). Buffers 264 and 274 may be a bus interface in the sense that they interface with the data bus, but they are not a bus interface for receiving requests to check cache coherency. Nevertheless, our own reading of Thacker shows a shared bus interface 280 which receives a request to check cache coherency and transmits a cache check request on line 278 to the logic 276 in cache 122 (col. 8, approx. lines 30-40).

The Examiner further finds (FR2) that the claimed "first[] read-only request to said cache" is providing address bits AB to comparator 262 and that the claimed "second read-write request that is forwarded to said cache if and only if said first read-only request indicates that said cache address tag matches an address of said request to check cache coherency and said request to check cache coherency requires said cache to be modified as a result of any of invalidation, copying out, or changing from clean to shared status" reads on providing address bits A to comparator 272.

This interpretation is in error for several reasons. First, the "first[] read-only request" and the "second read-write request" must go to the same cache, whereas the examiner finds the "first, read-only request" in first level cache 120 and the "second read-write request" in the second level cache 122.

Second, cache coherency checking is not performed in the first level cache 120. Data may be written to the first level cache 120 and, if it is, it is also written to the second level cache 122 (col. 7, lines 5-9). However, the first level cache 120 does not receive a request for checking cache coherency. Only the second level cache receives a request for checking cache coherency (from bus interface 280 via line 278; col. 8, lines 30-40, which is under the heading "CACHE COHERENCY" in col. 7). "The goal of the invention is to avoid accessing the first level cache for cache coherence checking, unless the updated location is actually stored in the first cache, because unnecessary coherence checking accesses to the first cache interfere with access by the processor and reduce the performance of the system." (Col. 2, lines 42-47). Thus,

the Examiner's reliance on the first level cache 120 is in error.

Third, while the second level cache 122 receives a request for checking cache coherency, it does not execute such request by a "first[] read-only request" followed by a "second read-write request" as claimed. The second level cache 122 checks cache coherency in the usual atomic read-write transaction manner. It can do this without hurting system performance by enabling bus buffer 290 to isolate the local bus 200 attached to the first level cache 120 from the local bus 200-1 attached to the second level cache 122 (col. 8, lines 14-22). Thus, the second level cache in Thacker does not perform as claimed in claim 1.

We have considered the Examiner's responses to Appellants' arguments (at EA4-5), but do not find them persuasive. As to point (1), the Examiner refers to the address compare operation as a "first snoop" and the write operation as a "second snoop" (EA4). The address compare in the first level cache is not in response to a request to check cache coherency and is not a read-only request in response to a request to check cache coherency. The write operation is

not a "read-write request" (emphasis added), but is only a write. As to point (2), the Examiner states that the avoidance of at least one write operation is not stated in the claims (EA4-5). This is true. However, since all cache misses will avoid a write operation and since cache misses will be a frequent occurrence (the specification indicates this will happen 90% of the time, page 10, lines 1-6), the advantage of the double snoop cannot be ignored. As to point (3), the Examiner finds that one piece of hardware cache performs an access twice in sequence (EA5). Our response to point (1) also applies here. The Examiner has not shown that Thacker teaches one piece of hardware cache that is accessed twice in sequence as claimed. As to point (4), in response to Appellants' argument that it would not have been obvious to perform a double snoop in Thacker, the Examiner states that the test for obviousness is not whether features may be bodily incorporated but simply what the reference makes obvious to one of ordinary skill in the pertinent art (EA5-6). This does not respond to Appellants' argument.

The Examiner further states (FR2):

> As to claims 1 and 6, Thacker et al. did not show each module including a clock for interrupting access to

> the cache during a read/write request. However, Thacker et al. show the timing for read and write data from a request module to a receiving module (see fig. 3A-B and col .7 [sic], lines 52-55). It would have been obvious for one of ordinary skill in the data processing art at the time the invention was made to incorporate a clock into each of the modules for keeping track of the requests from the module because it would improve the system taught by Thacker et al. by keeping track of the requests independently in each of the modules.

Appellants do not argue the "lock" limitation. We note that the Examiner has misinterpreted the claimed "lock for interrupting access" (emphasis added) as a clock. Therefore, the Examiner's conclusion that it would have been obvious to incorporate a clock does not respond to the claim limitation. It is clear that the Examiner is referring to a clock, instead of a lock, because the referenced text at column 7, lines 52-55, deals with clock lines. The Examiner's finding that figures 3A and 3B somehow deal with timing is not understood, since "FIGS. 3A and 3B show the allocation of address bits for two preferred embodiments of the present invention" (col. 3, lines 8-10). Figure 3C shows clock lines 116, but this has no relevance to the claimed lock. Since the Examiner's rejection fails to address the claimed "lock for interrupting access," the Examiner has failed to establish a prima facie case of obviousness for this additional reason.

While we see that bus buffer 290 in Thacker can interrupt access to the second level cache 122 during cache checking, the Examiner does not rely on buffer 290 or the second level cache.

In summary, the Examiner has failed to establish a prima facie case of obviousness with respect to the limitations of (1) performing a "first[] read-only request" followed by a "second read-write request" in response to a "request to check cache coherency," and (2) a "lock for interrupting access." Therefore, the rejection of claims 1, 2, 4-7, 9, and 10 is reversed.

REVERSED

LEE E. BARRETT Administrative Patent Judge	)	
MICHAEL R. FLEMING Administrative Patent Judge	)	BOARD OF PATENT APPEALS AND INTERFERENCES
ERIC FRAHM Administrative Patent Judge	)	

Records Manager
Legal Department, 20B0
HEWLETT-PACKARD COMPANY
P.O. Box 10301
Palo Alto, CA 94303-0890